AMENDMENTS TO THE SPECIFICATION

In the Specification:

Please replace the paragraph beginning on page 1, line 17 with the following amended paragraph:

--Computers operate under the control of a program consisting of coded, executable instructions. Typically, a program[[s]] is first written as a textual representation of computer-executable instructions in a high-level language, such as BASIC, PASCAL, C, C++, C#, or the like, which are more readily understood by humans. A file containing a program in its high-level language form is known as source code. The high-level language statements of the source code are then translated or compiled into the coded instructions executable by the computer. Typically, a software program known as a compiler is used for this purpose. The compiled form of a program is generally known as object code.--

Please replace the paragraph beginning on page 5, line 27 with the following amended paragraph:

--By way of example, the program elements of the language neutral representation 12 include objects instantiated to represent corresponding programmatic constructs of the compile unit. One or more objects, for example, are contained within a basic object corresponding to the compile unit. Typically, there is at least one object that represents a type declaration for a class within the compile unit. Similarly, the type class object may contain one or more member objects, such as a collection objects that represent statements and/or expressions. Type declaration objects further may be compartmentalized into one or more namespace objects. The ~~object-based~~ language neutral representation 12 thus arranges associated objects according to a defined hierarchy.--

Please replace the paragraph beginning on page 6, line 6 with the following amended paragraph:

--The ~~object-based~~ language neutral representation 12 facilitates transformation or conversion of the code represented by the programmatic elements into other target language implementations. By way of example, the target languages include high-level language implementations (*e.g.*, C++, C#, JAVA, or other source code) and/or lower-level implementations (*e.g.*, assembly, byte code). An object-based model, in accordance with the present invention, also facilitates manipulation, derivation, and/or modification of the respective objects.--

Please replace the paragraph beginning on page 6, line 13 with the following amended paragraph:

--A code generator 14 may be employed to transform the ~~object-based~~ language neutral representation 12 into a corresponding high-level language representation (*e.g.*, a textual, graphical, or symbolic representation) 16 in accordance with an aspect of the present invention. For example, the code generator 14 may utilize an interface (not shown) having associated interface components that enables the ~~object-based~~ language neutral representation 12 to be converted into the desired high-level representation of the code 16. By way of example, the interface components may include respective components to generate code for namespace objects, class objects, expression objects, statement objects, etc, which may form the compile unit. Each language into which the language neutral representation 12 is to be converted implements the code generator interface so as to enable the language-neutral representation to be converted into the particular language:--

Please replace the paragraph beginning on page 6, line 24 with the following amended paragraph:

--A converter 18 also may be employed to transform the language neutral representation 12 into a corresponding low-level language, executable or intermediate language representation 20 in accordance with an aspect of the present invention. The converter 18, for example, employs a compiler interface that enables translation from the ~~object based~~ language neutral representation 12 to the executable representation, such as assembly (or object code), byte code, or a desired intermediate language. The compile interface returns results of a compilation process, such as may include corresponding object code, an indication of any errors association with the compilation process, and/or a pointer or filename to the created assembly.--

Please replace the paragraph beginning on page 7, line 3 with the following amended paragraph:

--By way of further illustration, the converter 18 may include a language-specific compiler that implements an appropriate compiler interface to enable appropriate translation from the language neutral representation 12 to the executable representation 20. In particular, the compiler may implement an interface similar to the code generator 14 to convert the language neutral representation 12 into a desired language-specific representation 16. The converter 18 may then perform a conventional compilation to convert high-level representation 16 into the corresponding low-level code representation 20, such as assembly, byte code or an intermediate language.--